DOCUMENT RESUME

ED 208 483 CE 028 639

AUTHOR Brosley, Bud
TITLE Performance-Based Vocational Education in Post-Secondary Computer Technology Program. Final Report.
INSTITUTION Indiana Vocational Technical Coll., Muncie.
SPONS AGENCY Indiana State Board of Vocational and Technical Education, Indianapolis.
PUB DATE 30 Sep 80
NOTE 119p.; Not available in paper copy due to broken print.

EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS *Competency Based Education; Computers; *Curriculum Development; *Data Processing; *Data Processing Occupations; Instructional Materials; Job Analysis; Learning Modules; Occupational Information; Postsecondary Education; *Task Analysis; *Vocational Education

ABSTRACT

A study was conducted to develop a performance-based instructional program in electronic data processing (DP) and to identify the DP tasks within various occupational titles. In order to meet these objectives, task lists from various sources were sorted and merged into a final detailed list of tasks and subtasks, under the guidance of an advisory committee, resulting in the teaching unit, "Program Writing in Ten Tasks." A pilot study was conducted and found difficulty in identifying DP tasks within various job titles unless terms being used were further clarified; an empirical computer-based method for such clarification was suggested. The products of the study were the teaching unit and a pilot study of time-spent pattern similarity coefficients among job titles. The research study resulted in better quality and more uniformity in the teaching of programming courses, and better quality and more uniformity in defining of job titles used in the data processing industry. (Appendixes contain the teaching unit and the various job titles and task analyses.) (KC)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED204483

FINAL REPORT

Performance-Based Vocational Education in Post-Secondary Computer Technology Program

5/19/81

BEST COPY AVAILABLE

Bud Bromley

Project No. ~~VPP-III~~ 275-77-III-3

133

Indiana Vocational Technical College
Muncie, Indiana

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Joanna Bromley

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

September 30, 1980

State Board of
Vocational and Technical Education
401 Illinois Building
17 West Market Street
Indianapolis, Indiana 46204

CE 028 859

II. INFORMATION NEEDS

A. Population Included in Project

Type	ESTIMATED NUMBERS Involved in Project	Affected by Project
A Disadvantaged	A	A
B Handicapped (Jobless)	B 15	B
C Migrant	C	C
D Minority	D 3	D
E Combination of the above	E	E
F Non-English Speaking	F	F
G Other (Specify) Normal	G 32	G 300+

GROUP

1 Pre-school	1	1
2 Elementary	2	2
3 Junior High School	3	3
4 Middle School	4	4
5 Senior High School	5	5
6 Postsecondary	6	6
7 Adult	7	7
8 University	8	8
9 Employer	9	9
10 Employee	10	10
11 Citizens	11	11
12 Parents	12	12
13 Combination of the above	13 50	13 300+
14 Other (Specify)	14	14

LOCALITY (Check the one which encompasses the locality being studied)

a National	a	a
b State	b	b
c Region	c 50	c 300+
d District	d	d
e County	e	e
f Area	f	f
g Community	g	g
h Educational Agency (LEA/ER)	h	h

TABLE OF CONTENTS

III. Statement of the Objectives 1
IV. Statement of the Problem 2
V. Strategies 4
VI. Analyses Procedures 6
VII. Findings 9
VIII. Evaluation of Project 10
IX. Conclusions and Recommendations 13
X. Financial Statement 14

Appendices

Appendix A 15
Appendix B 43
Appendix C 44
Appendix D 45
Appendix E 47
Appendix F 51
Appendix G 53
Appendix H 57
Appendix I 59
Appendix I2 62
Appendix J 65
Appendix K 68
Appendix L 71
Appendix M 78
Appendix M2 85
Appendix N 95
Appendix O 98
Appendix P 100
Appendix Q 102
Appendix R 107
Appendix S 108

Appendix T 109
Appendix U 112

XII. Final Report Abstract 113

III. Statement of the Objectives

The original objective of this project was stated as follows: "The overall objective of this project is to develop a performance-based instructional program in Computer Programming Technology using V-Techs materials, and other task lists, as a resource for performance objectives." See Program Writing in Ten Tasks, Appendix A.)

A consultant to the project noted that the project objectives appeared premature in respect to the overall state-wide performance based vocational education strategies. The consultant suggested a revision to the objectives of the project. Those objectives were subsequently revised as follows: Tasks done by persons employed in various job titles in automated data processing would be identified. Then this information could be integrated into a more comprehensive performance based vocational education endeavor. That is, a computer program could identify the tasks unique to each job title and the task lists common to expanding heirarchies of job titles. (See Job Matrix, Appendix B.)

Statement of the Problem

Classroom teaching units now being used in computer technology classrooms are not yet known to be based upon tasks which are being done by people engaged in electronic data processing. Thus there is a need to develop learning packages which, by being based upon observable performances of people now in electronic data processing, will aid students in achieving realistic, and hence employable, competencies in Computer Programming Technology.

Furthermore, the learning packages need to be so organized as to represent an ascending sequence from the most general tasks to the most specialized tasks. This implies the prior need to know how these tasks occur within the job titles now being occupied in various industries employing electronic data processing personnel. Specifically, the need is to know:

1. what job titles occur in the various industries. (See Categories of Businesses, Appendix C.)
2. within each job title surveyed, what tasks are being done. (See Job Titles in Each Interviewed Business, Appendix D, and Identified Tasks and Skills, Appendix E.)
3. what percent of the total job title time is taken up by each task.

Certain constraints upon the problem were set fairly early in the project, by agreements of the participants and the consultants. It was agreed first that word processing would not be considered for this project; i.e, the priority was to be on data processing. Second, it was decided to emphasize the data processing itself as opposed to supervisory, managerial, or clerical activities peripheral to the data processing. Third, it was decided to work within the area of electronic data processing rather than all of

automated data processing because the latter includes areas of mechanical data processing which are now obsolete.

While these decisions were well within those implied by the objectives, the further rationale for them was that the participants could bring more experience to bear on the selected areas than on those which were not chosen.

V. Strategies

In pursuance of the original overall objective, two strategies were concurrently initiated:

1. Within the rapidly burgeoning field of electronic data processing, it was necessary to converge to some unit of the instructional program of a reasonably achievable size.
2. To keep the instructional unit performance-based, as mandated, it was necessary to have it demonstrably based upon task lists, both from V-Techs lists and other task lists, selected to conform to the chosen instructional unit.

The evolving of the first strategy may be seen in the sequence of appendices: F, G, H, and I. The direction of this strategy was influenced by the growing prevalence of small computers in advertisements, in stores, and, most importantly, in small businesses. Its major steps were:

1. a rather general activity analysis.
2. a somewhat more program-oriented analysis pointing up the relevance of programming knowledge.
3. a narrative description of program development.
4. a more operationally oriented activities description of program development.

The second strategy may be noted in appendices J, K, L, and M. The major steps here were to:

1. scan V-Techs for DP tasks.
2. scan the Region 8 lists for DP tasks.
3. merge the two lists.
4. delete those tasks clearly at a managerial and/or supervisory level, and those of a clerical nature.

(borderline or questionable items were retained for further consideration by committees.)

5. classify tasks as being:
   a. "essential" to "not needed".
   b. school teachable vs. on-the-job training.
   c. within the scope of an Associate degree program or not.
   d. DP oriented vs. managerial/supervisory or clerical (for the borderline tasks).
6. group the tasks into larger categories.

These two strategies tentatively converged, as shown first in Appendix N, and finally crystalized into Program Writing in Ten Tasks (Appendix A.)

As this unit of work emerged, it became apparent that there was not yet enough information to locate the unit among other potential similar units. This observation led to the revision of the objectives (q.v.) which gave rise to the last strategy. As recommended by the initiating consultant, the steps were to:

1. ascertain job titles occuring in the varying industries. (See Businesses Interviewed, Appendix O.)
2. find the tasks actually being done by interviewing persons in various job titles.

... Analysis Procedures

The analytical prodedures for the first part of the project are given, in essence, in the two concurrent strategies (q.v.). The principal investigator offered suggestions which were reacted to by committee members. Discussion produced a concurrance of opinion which then served as the starting point for the next step. The Program Writing in Ten Tasks, which resulted from these steps, is the analysis of each of the ten tasks.

For the revised objective the data was a printout of DP job titles, as reported by returns of questionnaires sent to industries statewide in a project directed by Pat Sahacki, Indiana V-Tech Consultant. The original printout of eighteen pages contains nine hundred detail lines. Each line contains a job title, the number of people employed in that job title by a responding industry, and the percent of total work time spent by them in each of five categories: Systems and Design, Programming, Machine Operations, Control, and Others. This material is summarized in Appendix P. However, the job titles received from various industries were not always consistant, as evidenced in the Annotated Listing, Appendix Q.

These comments are not criticism, neither of that project nor of the cooperating industries. It is much more likely that the lack of stable terminology is due to the turbulent growth of electronic data processing in many industries. But it does reveal a present problem which must be dealt with.

For specific examples, consider the time-spent patterns listed for eight "Analyst/Programmers" and seventeen "Programmer/Analysts", as shown in Appendix R. Cases 1, 4, 9, 17, 23, and 24 have identical time-spent patterns although

two are listed as Analyst/Programmers, and six as Programmer/Analysts. This alone is acceptable; the titles are really quite similar. But now note that cases 3 and 7 have drastically differing time-spent patterns, although both are listed as Analyst/Programmers. Such instances found at random suggest the need for a more systematic examination of the similarity, or dissimilarity, of the time-spent patterns reported from different industries for their various job titles.

Cattell's coefficient of pattern similarity was chosen to report the resemblances among the time-spent patterns reported for job titles rather than the more familiar coefficient of correlation, for, as Cattell notes, the coefficient of correlation could give falsely high results in either or both of two situations.

Should all the elements of a first pattern be greater by a constant amount than all the elements of a second pattern, the coefficient of correlation would be 1.00 even though the first set of elements would indicate a quantitative difference from the second set.

The other condition giving a falsely high coefficient of correlation (falsely, that is, for purposes of pattern comparisons) is that if the highs of the first pattern should be higher than those of the second pattern and the lows of the first be lower than the lows of the second, then the correlation would also be too high.

These are, of course, the geometric analogs of the fact that a coefficient of correlation is independent of any linear transformations of its variables. While this characteristic is valuable in numerous statistical situations, it can obscure real differences among sets of data which happen to consist of patterns which need to be

compared.

In the situation where the time-spent patterns are expressed as percents of the total time, the total for each pattern should be 100%, and no one pattern could be consistantly higher than any other. But the second condition, in which the elements of one pattern may be more extreme, both high and low, could occur, and would be falsely indicated by the use of a coefficient of correlation.

See Appendix S for a brief description of the statistic. See also Cattell, Raymond B., R_p and Other Coefficients of Pattern Similarity, Psychometrika, 1949, Vol. 14, p 279-298.

A pilot study was run by getting all three hundred pattern similarity coefficients among the twenty five cases listed in Appendix R. The resulting matrix of pattern similarity coefficients is given in Appendix T. (In T, there are six hundred values because, for ease of reference, both $r_{i,j}$ and $r_{j,i}$ are given, though they are equal.)

Coefficients range from r = 1.00 for twenty six pairs to r = -0.49 for cases 3 and 15. Of the twenty six coefficients of 1.00, fifteen are among one cluster, cases 1, 4, 9, 17, 23, and 24. Another ten are among a second cluster, cases 14, 16, 18, 19, and 20. The last 1.00 is for cases 6 and 12 (see Appendix U). Cases 7 and 15 are very similar (r = 0.98) though not much like any other cases. Cases 3 and 8 bear no similarity to each other (r = 0.00) and little similarity to any other cases. Found here, more rigorously than earlier, are similarly named time-spent patterns that are rather different; and differently named time-spent patterns that are rather similar.

VII. Findings

Program Writing in Ten Tasks is the output of the first part of the project, and as such constitutes the findings of that part. See Appendix A.

For the revised objective, the major finding at this point is only tentative, but it would appear to be that the jobs themselves are not clearly identified by the job titles given to them by the various industries answering the questionnaire.

One other finding may be cautiously suggested by the pilot study described in the Analysis Procedures: It may be possible to find closely related jobs, to which it would be reasonable to apply one job title, by an analysis of their patterns of time spent at various tasks, and regardless of the names given to them in various industries. Cattell's coefficient of pattern similarity is one possible statistic to use. However, the use of cluster analysis on larger matrices is open to question.

VIII. Evaluation of Project

The rigorous evaluation of Program Writing in Ten Tasks has not been done; the unit is still open to revisions by comments from students, teachers, and DP professionals to whom it has been submitted. Though subjective, the comments so far on the present form have been generally favorable. Several students asked, "Why didn't we get this sooner?" Instructors have noted that it contains, compactly, many things they hope not to forget while teaching a programming language.

Several aspects of the pilot study cited in the Analyses Procedures deserve evaluative comment.

The reasons for using Cattell's coefficient of pattern similarity have already been given in the Analyses Procedures. However, there is also an argument favoring the use of the coefficient of correlation. The technique of physically looking for clusters of high coefficients within a matrix of coefficients, while useful in a small pilot study such as was done here, is not suited to larger projects, for several reasons; the technique of factor analysis, which starts with a matrix of correlations, is better.

For one thing, if the time-spent patterns were based upon several dozen items per job title, rather than just five, there would be little probability of coefficients of 1.00, and the highest similarities would be much harder to see.

Furthermore, finding clusters is not an exact technique. We can find Cluster I and Cluster II, and note the pair, cases 6 and 12 (Appendix T), but we have no indication of how well each one represents the data of the whole matrix. But if we extract the first centroid factor, we find that the

6-12 pair comes closer to the centroid of the whole matrix than do any other cases. (Their Factor I loading is 0.97. While iterations would probably not change the top rank position, rotation for simple structure undoubtedly would.)

Lastly, had the matrix been made for all nine hundred detail lines, it would have contained 404,550 coefficients in one triangle, or 809,100 coefficients in all, rather much for a human being to search through.

For at least these reasons, the technique of factor analysis (either centroid or principal components), for which there are already excellent computer programs, would appear more useful for finding time-spent patterns which are sufficiently alike to justify assigning one job title to each group. But the mathematical derivation of factor analysis starts from the assumption that the first matrix is a matrix of Pearson correlations, not a matrix of pattern similarity coefficients.

But Cattell specifically designed the coefficient of pattern similarity to resemble the coefficient of correlation, and since he did, it might quite possibly be that his pattern similarity coefficient is sufficiently like the correlation coefficient so that a factor analysis of a matrix of pattern similarity coefficients might be at least pragmatically useful, if not mathematically elegant.

The last evaluative comment concerns terminology. One consultant suggested that Program Writing in Ten Tasks actually contains three or four domains, such as inputting, processing, outputting, and possibly, debugging. At the other extreme, another consultant felt that the whole unit is but one single task. Here again, it may be that the application of established definitions to this rapidly changing area of vocational interest may be less reliable as

yet than to other more accepted and stabilized vocational areas.

IX. Conclusions and Recommendations

At present, it seems safe to conclude that Program Writing in Ten Tasks is worthy of further trial usage. For example, the first listing of the ten tasks could be a hand-out for consideration in a class on the fundamentals of data processing. The analyses of the first four tasks should prove appropriate to classes in problem solving and flowcharting. The analyses of the last six tasks would then be used in classes on specific languages.

It is recommended that further study be directed toward a concensus of expert opinion as to just what does constitute jobs, domains, tasks, and subtasks in the program writing area of electronic data processing.

The pilot study (see Analyses Procedures) does appear to offer some recommendations for a possible empirical clarification of terms. In the pilot study, the job title time-spent patterns were reported in just five large categories. The analysis could be improved if, for each job title, the time spent could be reported for each of several dozen smaller units of activity (e.g. tasks), each defined as explicitly as possible. Thus, each coefficient of pattern similarity would represent a more accurate measurement of similarity than do those in the pilot study.

In the pilot study, only twenty five cases--job titles--were analysed by a microcomputer to produce the matrix of pattern similarity coefficients. For a full scale study, a larger computer could work with a much larger group of job titles, and could contain the resultant similarity matrix. The factor analysis of this matrix could be run to see if the results would appear reasonable to practicing experts in the field of electronic data processing.

FINANCIAL STATEMENT FOR FINAL Fiscal REPORT ENDING September 30, 1980
(Date)

Project No. 275-78

ITEMS	Project Budget FY 19--		Total Expenditures on Project		End of Project Balance	
	Agency	Federal	Agency	State/ Federal	Agency	State/ Federal
A. DIRECT EXPENDITURES						
1. Personnel	$3,900.00	$22,500.00	$3,900.00	$16,815.42	-0-	$ 5,680.58
2. Contractual services		9,010.00		2,125.11		6,884.89
3. Employee benefits		490.00		122.29		367.71
4. Travel		1,000.00		189.78		810.22
5. Supplies and materials		500.00		122.44		377.56
6. Communications (includes phone calls, printing)		500.00		408.20		91.80
7. Properties (rentals or purchase of equipment)						
8. Facilities	1,104.00		1,104.00		-0-	
9. Product production and dissemination		1,000.00		37.50		962.50
10. Project Evaluation						
B. INDIRECT EXPENDITURES						
C. TOTAL EXPENDITURES	$5,004.00	$35,000.00	$5,004.00	$19,824.74	-0-	$15,175.26

Fiscal Agent/Director: John T. Hatchett
John T. Hatchett, Vice President/Treasurer

Institution/Agency Name: Indiana Vocational Technical College

Refund this amount to the State Board of Vocational and Technical Education $6,425.26

X. FINANCIAL STATEMENT

14

12

20

XI. APPENDICES

APPENDIX A

PROGRAM WRITING
IN TEN TASKS

INDIANA VOCATIONAL TECHNICAL COLLEGE

PROGRAM WRITING
in ten tasks

Step-by-Step analyses of each task

These materials are still in the developmental stage. We would welcome any comments you might care to make.

Dr. Jack J. Bainter, Project Director
4100 Cowan Road, Muncie, IN 47302
Attn: C. Bud Bromley

Task 1: From the boss's description of some computer work he wants done fill out a printer spacing chart showing the desired output.

Task 2: Specify what input is required* in order to get the desired output.

Task 3: Specify how each input datum is to be coded for input into the program.

Task 4: Specify what ordered steps (procedures, algorithms) are needed to get the desired output from the required input.

Task 5: Create (utilize, modify, originate, and/or combine) a program in a high level language to execute the ordered steps necessary to produce the output.

Task 6: Create test data to test every anticipatable variation in the real data the program is intended to accept.

Task 7: Enter program and data, and request execution of the program. (Other combinations are possible--e.g., enter program without data, and request compilation only--but will not be considered within this project.)

Task 8: Detect and correct compiler-noted errors.

Task 9: Detect and correct all non-compiler errors.

Task 10: Document the finalized program.

*A few programs require no input data. However, all programs should output. (It is possible to have unprinted outputs, such as outputting a file to a disk, but that is beyond the scope of this project.)

TASK ANALYSIS GUIDE

Task Statement:

From the boss's description of some computer work he wants done, fill out a printer spacing chart showing the required output.

Task Number

1

Task Performance Cues:

The boss has described an output which he wants.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
1. Items in sequence:				
a. Has the boss specified exactly what report titles he wants?	4	2		Some of these items do not apply to some kinds of outputs.
b. Has he specified what column headings he wants?	4	2		
c. Has he said whether or not he wants pagination?	4	2		
d. Has he specified what control breaks he wants?	4	2		
e. Has he specified what the key field is for each control break?	4	2		
f. Has he specified what items of the data are to be group indicated for each control break?	4	2		
g. Has he specified what minor totals are to be printed for each control break?	4	2		

Task elements — Task 1	Decisions: Yes	No	Go To Step	Comments
1. (continued)				
n. Has he specified the contents of all detail lines that he wants?	4	2		
i. Has he specified what items of the data are to be group printed?	4	2		
j. Has he specified what final totals he wants?	4	2		
k. Has he specified what numbers are to be zero suppressed?	4	2		
l. Has he specified where zero balances are to be shown?	4	2		
m. Has he specified where to have floating dollar signs?	4	2		
n. Has he specified data to be flagged?	4	2		
o. Has he specified any other kind of editing and/or special marking?	4	2		
p. Has he specified any comments?	4	2		
q. End of sequential items.			0	
2. Can you, from experience, supply the needed details?	4	3		
3. Ask him for the needed details.			4	
4. Start a variables name list. For each variable required, devise a unique variable name, specify its attributes (e.g., integer, real, complex, vector, matrix; alphabetic, numeric, logical; width of field, position of decimal point; internal storage mode, precision), and tell briefly what it does in the program. To describe a constant being used, give its value, attributes, and use in the program.			5	Variable names must agree with the rules of the language. Languages differ in the attributes which must be specified. Other variables to be added later should be described in the same way.

Task Analysis Guide

Task Elements	Task 1	Page 3 of 3 Pages	Decisions Yes	No	Go To Step	Comments
5.	Return to the next item in sequence.				1	
6.	Fill out a printer spacing chart.				7	
7.	Does it satisfy the rows in every detail?				EXIT	4

TASK ANALYSIS GUIDE

Task Statement:

Specify what input data is required* in order to get the desired output.

Task Number 2

Task Performance Cues:

A printer spacing chart has been completed showing the desired output.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
1. Does the program require input data?	2	EOF		
2. Is some of the required output numeric?	3	7		
3. Are some of the output data the results of calculations?	4	7		Item 3 includes counters and accumulators.
4. Do you know the necessary formulae?	6	5		
5. Consult suitable references (human or printed).			6	*A few programs require no input data. However, all progams should output. (It is possible to have unprinted outputs, such as outputting a file to a disk, but that is beyond the scope of this project.)
6. Add, to yourvariable name list, each variable and constant used by the formulae.			7	

TASK ANALYSIS GUIDE

Task Elements	Task 2	Page 2 Of 2	Decisions Yes	Decisions No	Go To Step	Comments
7.	Are some of the output data the results of decisions?		8	11		
8.	Do you know the variables which are needed to determine what way each decision will go?		10	9		
9.	Consult suitable references.				10	
10.	Add these variables to your variable name list.				11	
11.	Do you know which data are to be output from the input data without modification?		13	12		
12.	Consult suitable references.				13	
13.	Do you know which data need to be modified for outputting?		15	14		
14.	Consult suitable references.				15	
15.	Do you know what variables are needed to modify the output data?		17	16		
16.	Consult suitable references.				17	
17.	Add these variables to your variable name list.				18	
18.	Do you know whether or not any other variables are needed in the program?		20	19		
19.	Consult suitable references.				20	
20.	Add these variables to your variable name list.				EOT	

TASK ANALYSIS GUIDE

Task Statement:

Specify how each input datum is to be coded for input into the program.

Task Number 3

Task Performance Cues:

The data to be input has been established.

Task Performance Procedures:

A. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
				"Coding" is not "encryption"; encrypting is beyond the scope of the project.
1. Are there specific coding instructions already supplied?	2	4		There might be shop usages, or specific user instructions.
2. Follow the instructions.			3	
3. Do the instructions cover all the data being input?	15	4		
4. Are there verbal data having measurement attributes, an order of some sort (a gradient, a sequence, or an established arbitrary order)?	5	7		e.g. "Yes, No, Uncertain"
5. Assign to each such word a number which represents its position in the order. Reserve zero for responses not in the order.			6	Zero might be used for "not applicable," "unknown", or for blanks.

TASK ANALYSIS GUIDE

Task Elements	Task 3	Page 2 Of 2 Pages	Decisions Yes	No	Go To Step	Comments
6. Write a list of each word and its code number.					7	
7. Are there verbal data which could be more compactly represented by numbers?			8	14		e.g. 20-letter stock part names might also have 8-digit part numbers.
8. Would encoding jeopardize any of the desired verbal output?			9	10		
9. Does the boss choose to use encoding?			10	14		
10. Are there numbers already attached to the verbal data?			11	12		
11. Use the existing numbers.					13	
12. Devise a numbering system.					13	
13. Write a list of each word and its code number (or give a reference to such a list).					15	
14. Use the data uncoded.					15	
15. Complete the listing of how all data are being handled.					EOT	

TASK ANALYSIS GUIDE

Task Statement:

Specify what ordered steps (procedures, algorithms) are needed to get the desired output from the required input.

Task Number
4

Task Performance Cues:

The coding of input data has been established.

Task Performance Procedures:

A. when a step is completed, go to the next indicated step.

B. when a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
				Tasks 4 and 5 may need to be repeated as a result of step 12 of task 6.
1. Do you know how to produce the output from the input?	4	2		
2. Consult suitable references (human or printed).			3	
3. Devise procedures when necessary.			4	
4. With simple data, go thru the steps from input to output, listing, in the order in which you do them: a. any variables or constants which you input. b. any variables you make use of. c. any constants you make use of. d. any counters you make use of. e. any accumulators you make use of. f. any formulas you make use of. g. any new variables you generate. h. any decisions you must make, and the variables which control them.				"Variables" is here intended to include all arrays, structures, or other nameable groupings of variables.

Task Elements	Page 2 Of 3 Pages	Decisions Yes	Decisions No	Go To Step	Comments
Task 4					
i. any branching you do, and the variables, if any, which direct the branching.				5	
j. any variables or constants which you output, and the specifications needed to control their editing and placement format.					
5. Are there sets of steps which are repeating (cyclical, iterative, a loop)?		6	8		
6. Specify the conditions which start each loop.				7	
7. Specify the conditions which end each loop.				8	
8. Are there variables which would be more easily dealt with as a group (an array, a structure, in levels)?		9	10		
9. Group them, and revise the name list accordingly.				10	
10. Consider each numeric value. Are there any which should not go beyond (either above or below) some reasonable limit?		11	19		
11. Do you want the program to output a warning whenever the limit is passed?		12	15		
12. Devise steps which will: a. output the warning whenever the limit is passed, or b. output the warning after the regular output is completed, or c. output the warning on another output device. Add these steps to your ordered list.				13	
13. Do you require any new variables or constants for the limits you have set?		14	15		

Task Elements — Task 4	Page 3 OF 3 Pages	Decisions Yes	Decisions No	Go To Step	Comments
14. Add these to your list of defined variables and constants.				15	
15. Do you wish to process the beyond-limits data as is (i.e., without revision)?		19	16		
16. Do you wish to revise the beyond-limits data, then process them?		17	18		
17. Devise steps to cause the desired revisions, and add the steps to your ordered list.				19	
18. Devise steps to bypass the beyond-limits data and continue the program; add to list.				19	
19. Are there any pairs of variable names such that the second is not called into use for the first time until after the first has been used for the last time?		20	22		
20. Would it be helpful to use one variable name for both variables in each case?		21	22		
21. Devise a name, and revise the variable name list to describe correctly the dual-purpose variable name, for each such name.				22	
22. Does your revised ordered list of steps still in fact produce desired output?		BUT	1		NOTE: The ordered list of steps may be a flowchart, pseudocode, a listing similar to this one, or any other unambiguous style capable of showing looping.

TASK ANALYSIS GUIDE

Task Statement:

Create (utilize, modify, originate, and/or combine) a program in a high level language to execute the ordered steps necessary to produce the output.

Task Number 5

Task Performance Cues:

An ordered list of steps to produce the output has been established.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
1. Are there any programs (or subroutines) already available which claim (see their documentation) to do some, most, or all of what you want?	2	9		
2. Go thru the program (subroutine) and list each of the steps it does in the order in which it does them.			3	
3. Compare this second list with your list of ordered steps.			4	
4. Are there blocks of steps in both lists which are the same?	5	6		
5. Copy out of the supplied program the statements which produce the steps you want; these become part of your program.			6	
6. Are there blocks of steps in both lists which are at least somewhat similar?	7	9		

Task Elements	Page 2 Of 4 Pages	Decisions Yes	Decisions No	Go To Step	Comments
	Task 5				
7.	Can you modify the statement so as to produce correctly the steps you want?	8	12		
8.	Add the modified statements to your program, in correct order.			9	
9.	Are there any remaining steps (or blocks of steps) in your first list for which you have no matching programs or subroutines?	10	35		
10.	Can you write routines in your high level language for each of these steps?	11	12		
11.	Add these statements to your program, in correct order.			9	
12.	Is this step one of: (items in sequence)				
	a. inputting?	14	13		
	b. storing?	14	13		
	c. outputting?	14	13		
	d. formatting?	14	13		
	e. organizing data (e.g., sorting merging, ordering, setting up tables, updating)?	14	13		
	f. calculating?	14	13		
	g. deciding?	14	13		
	h. branching?	14	13		
	i. starting a loop?	14	13		
	j. ending a loop?	14	13		
	k. other modifications of data (e.g., editing)?	14	13		
	l. setting control variables for one of the steps above?	14	13		
	m. The step is none of the above.			26	
13.	Return to the next item in sequence.			12	
14.	Review all the statements (commands) in your language which deal with this activity.			15	
15.	Pick a possibly useful statement; note which parts are fixed (unchangeable) and which are variable (you can put in whatever you want).			16	

TASK ANALYSIS GUIDE

Task Elements — Task 5	Page 3 Of 4 Pages	Decisions Yes	Decisions No	Go To Step	Comments
16. Consider various combinations of changes of the variable parts. Eliminate those which violate the language rules, and those which clearly violate the logic of your steps.		17			
17. Consider the effect of each possible remaining variation.				18	
18. Does it execute the step you want?		19	20		
19. Add it to your program, in correct order.				9	
20. Have you tried all reasonably possible variations of a statement?		21	17		
21. Consider various orders of several statements at a time. Eliminate those which violate the language rules, and those which clearly violate the logic of your steps.				22	
22. Consider the effect of each of the remaining possible orders of several statements.				23	
23. Does one of these orders execute the step (or steps) you want?		24	25		
24. Add the steps of it to your program, in correct order.				9	
25. Have you tried all the reasonably possible orders of the statements?		26	22		
26. Reconsider your original list of steps.				27	
27. Could any of the steps be changed without degrading the quality of the output?		29	28		
28. Could the order of any set of several steps be changed without degrading the quality of the output?		29	32		

Task Elements — Task 5	Page 4 Of 4 Pages	Decisions Yes	Decisions No	Go To Step	Comments
29. Revise your original list accordingly, and revise your list of variable names if necessary.				30	
30. Can you now find or write routines in the high level language to do the revised steps?		31	32		
31. Add them to your program, in correct order.				9	
32. Consult with other programmers.				33	
33. Can you now write the routines to do the steps?		31	34		
34. Start again, considering wider variations of any and everything that can possibly be varied than you considered the first time.				1	
35. Check thru your program. Do you need any housekeeping statements (such as reserving storage, defining variable types, opening files or initializing variables)?		37	36		
36. Do you need any wrap-up statements (such as outputting final counts, final totals, closing files, or outputting an EOJ indicator)?		37	EXIT		
37. Add any such statements to your program, in correct order.				36	

37

TASK ANALYSIS GUIDE

Task Statement:

Create test data to test every anticipatable variation in the real data the program is intended to accept.

Task Number 6

Task Performance Cues:

The first draft has been produced of the required program in a high level language.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
1. Devise a small amount of input data typical of that which the program is intended for. This is the start of your test data.			2	
2. Consider each decision in the program, in order.			3	
3. For a decision, consider each possible branch (i.e., each way it can go).			4	As used here, "decision" includes all conditional branching.
4. For each possible branch, devise a small amount of test data which will cause the program to take that branch; add it to your test data.			5	
5. Has data been devised for all decisions?	6	2		
6. Consider, in turn, each field being input.			7	

TASK ANALYSIS GUIDE

Task Elements — Task 6	Page 2 of 2 Pages	Decisions Yes	Decisions No	Go To Step	Comments
7. Are there any reasonably possible variations (including rare but not unreasonable variations) which are not yet in the test data?		8	9		Unreasonable variations were dealt with in task 4, starting at step 10.
8. Devise data for each such variation, and add them to your test data.				9	
9. Might the boss know of yet other possible variations?		10	12		
10. Consult him. Has he any additions?		11	12		
11. Add them to your test data.				12	
12. Will your program now handle correctly all the test data you now have?		14	13		
13. Repeat Tasks 4 and 5.				12	
14. Independently of the computer, calculate the output which should result from the correct processing of the test data.				EXIT	

TASK ANALYSIS GUIDE

Task Statement:

Task Number 7

Enter program and data, requesting execution of the program.

Task Performance Case:

Both program and test data are completed. (The program may, however, need revision.)

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
1. Is your system keyboard to CRT?	2	8		
2. Enter a line of the program (or data) and proofread it.			3	
3. Is it correct?	7	4		
4. Do you know how to correct it?	6	5		
5. Consult suitable references.			6	
6. Make the necessary corrections.			7	
7. Is this the last line of the data?	14	2		
8. Are you keypunching cards?	10	9		
9. Enter the program data by your system.			14	
10. Keypunch all the cards needed for program and data.			11	

Task Elements	Task 7	Page 2 of 2 Pages	Decisions Yes	Decisions No	Go To Step	Comments
11. Proofread or verify them.					12	
12. Are there any errors?			13	14		
13. Keypunch correct cards.					14	
14. Do you know how to compile and execute the program?			16	15		
15. Consult suitable references.					16	
16. Submit the program and data to the CPU for compiling and executing.					17	
17. Does the output include at least a listing of the program?			EOT	14		

TASK ANALYSIS GUIDE

Task Statement:

Task Number 8

Detect and correct compiler-noted errors.

Task Performance Cues:

The output includes at least a listing of the program.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
				Other errors will be considered in task 9.
1. Has the compiler indicated errors?	2	AUT		Compilers differ rather widely in the ways they output messages.
2. Do you understand the error messages?	4	3		
3. Consult suitable references.			4	
4. Do you see what elements in your program are unacceptable to the compiler at each indicated error?	5	6		
5. Correct each one.			10	
6. Can you correct any of the indicated errors?	7	8		
7. Make all the corrections you can.			10	
8. Recheck the suspect parts of your program, an item at a time, by: (items in sequence)				
a. More carefully proofreading for erroneous characters.				
b. Rechecking each part of every high level statement for its agreement with the rules of the language.			9	Experience will sharpen the student's ability to be correctly suspicious.

TASK ANALYSIS GUIDE

Task Elements — Task 8	Page 2 Of 2 Pages	Decisions Yes	Decisions No	Go To Step	Comments
8 cont.					
c. Rechecking your nigh level language statements against your ordered list of steps to see if the statements are in fact actually doing exactly what you really want, in every detail.				9	
d. Checking every variable to see that it is the right type or mode ("field definition", "format", "picture") and the right size.				9	
e. Tracing forward every generated variable to verify that it is defined correctly for its following uses.				9	Variables are defined either by being input or by being generated by an assignment statement.
f. Tracing backward every variable called for to verify that it has been previously defined correctly.				9	
g. Rechecking the logic of the suspect parts, including all branching into and out of them.				9	
h. End of items in sequence.				11	
9. Does this bring to your attention a correction to be made?		10	8		
10. Does this complete all the corrections you need to make?		12	4		Sometimes one correction will clear several compiler messages.
11. Consult suitable references on the particulars of your compiler.				9	
12. Rerun the corrected program.				1	

TASK ANALYSIS GUIDE

Task Statement:

Task Number 9

Detect and correct all non-compiler errors.

Task Performance Cues:

The compiler has output a program listing which is free of compiler-noted errors.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
				Depending on the OS and compiler some warnings should be regarded as errors.
1. Does the output agree in format, in every detail, with the printer spacing chart produced in task 1?	7	2		
2. Can you, from clues suggested by the errors, locate and correct the program statements causing the faulty formatting?	6	3		
3. Locate each program statement which affects output formats in any way.			4	
4. Within each statement, locate each option which is set by the programmer.			5	Errors in the required parts of statements were corrected in task 8.
5. Can you see any option which might be changed in any way so as to correct the output?	6	35		

TASK ANALYSIS GUIDE

Task Elements	Task 9	Page 2 Of 4 Pages	Decisions Yes	No	Go To Step	Comments
6. Make the revision in the option.					30	
7. Do the data values output agree, in every detail, with the values which were independently calculated in task 6, step 14.			20	8		Be flexible enough to consider a wide range of unusual possibilities.
8. Recheck your input test data.					9	
9. Is each datum exactly the value you specified in task 6, step 14?			11	10		
10. Correct the faulty data.					30	
11. Check all program statements which affect data inputting.					12	
12. Is each datum being input according to the attributes assigned in Task 2?			14	13		
13. Correct the data and/or statements to get the right input.					30	
14. Have all accumulators, counters, switches, and/or indicators been properly initialised?			16	15		The initialising required varies with the language.
15. Correct the initialising.					30	
16. Check every step (procedure, decision, branch) which affects the values being output.					17	
17. Does each program step do in fact what your ordered list of steps requires?			19	18		
18. Make the corrections.					30	
19. Have you succeeded in finding the corrections needed?			20	35		
20. Has the system given any other error or warning messages (i.e., is the run terminating abnormally)?			21	27		The details in these items are more OJT than classroom items.
21. Do you understand the messages?			23	22		
22. Consult suitable references.						

Task Elements — Task 9	Page 3 OF 4 Pages	Decisions Yes	Decisions No	Go To Step	Comments
23. Can you make the required revisions?		26	24		
24. Consult suitable references.				25	
25. Can you now make the required revisions?		26	35		
26. Make the revisions.				30	
27. Have you made any changes in the program in <u>this</u> run thru this Task?		Task 8 Step 1	28		Completing the corrections may require several runs through this task. See steps 35 and 36.
28. Are there any temporary outputs which, having served their purpose, should be removed?		29	EOT		
29. Remove them; check each off the list.				Task 8, Step 1	
30. Do your changes make it necessary to revise the list of ordered steps made in Task 4?		31	32		
31. Revise the list of steps.				32	
32. Do your changes make it necessary to revise the list of variables started in Task 4, step 4?		33	34		
33. Revise the variables list.				34	
34. Return to the next numbered statement after the one that sent you to 30.				N+1	
35. Into the suspect portion of the program, insert temporary output statements to output intermediate variables, such as data variables, branching control variables, decision variables, switch and indicator variables.				36	

TASK ANALYSIS GUIDE

Task Elements — Task 9	Page 4 Of 4 Pages	Decisions Yes	Decisions No	Go To Step	Comments
36. Make a list of all such temporary output statements.				37	If necessary steps 35-38 may need to be gone through several times closing in on the trouble by the binary search technique.
37. From the calculations of Task 6, step 14, find out exactly what each temporary output statement should output.				38	
38. Mark the correct output for each temporary statement.				Task 8, Step 1	

TASK ANALYSIS GUIDE

Task Statement:

Task Number 10

Document the corrected finalized program.

Task Performance Cues:

The program runs without error, producing error-free output from the test data.

Task Performance Procedures:

A. When a step is completed, go to the next indicated step.

B. When a decision must be made, go to the step indicated by "Yes" or "No."

Task Elements	Decisions Yes	Decisions No	Go To Step	Comments
				If there are established shop practices for documenting, follow them.
1. Make a professional quality copy of the printer spacing chart produced in task 1.			2	
2. Make a professional quality copy of the variable name list started in task 1.			3	
3. Does the program require input data?	4	6		
4. Make a professional quality copy of the multiple card layout form (or equivalent form) made in task 3, step 16.			5	
5. Make a professional quality copy of the description of the data encoding completed in task 3, step 15.			6	
6. Make a professional quality copy of the flowchart (or equivalent form) produced in task 4.			7	
7. Make a professional quality copy of the test data devised in task 6.			8	

Task Elements — Task 10	Page 2 Of 2 Pages	Decisions Yes	Decisions No	Go To Step	Comments
					Steps 9-11 should include a list of all halt instructions and any other special instructions to the computer operator.
8. Write a concise description of the purpose of the program.				9	
9. Write a concise description of the procedures (the algorithms and decisions i.e. "how it runs").				10	
10. Are there any unusual procedures?		11	12		
11. Describe all unusual procedures in more detail.				12	
12. Write, for the end-user, a concise description of the steps gone through to use the program.				13	
13. To the source program, add comments which will help the language-knowledgeable user to relate blocks of the program to the flowchart, the list of variables, and the descriptions.				14	
14. Get a printout of the finalised and commented program.				15	
15. Get a printout of the output resulting from running the test data.				16	
16. Combine these documents into one neatly organised task.				EJT	

APPENDIX B

JOB MATRIX

A	B	C	D	E	F	G	H
Automated Data Processing (not Word Processing)	Systems	Systems and Designs	Supervisor Analyst Analyst Trainee Project Leader	Tasks specific to each job	Tasks common to Systems and Designs	Tasks common to Systems	Tasks common to Automated Data Processing
		Programming	Supervisor Junior Programmer Staff Programmer Programmer Trainee	Tasks specific to each job	Tasks common to Programming		
	Operations	Machine Operations	Supervisor Data Entry	Tasks specific to each job	Tasks common to Machine Operations	Tasks common to Operations	
		Control Operations	Supervisor I/O Control Operator Tape Librarian	Tasks specific to each job	Tasks common to Control Operations		

APPENDIX C

Categories of Businesses

1. Airports
2. Construction
3. Finance
4. Institutions
   a. Colleges/Schools
   b. Governmental
   c. Hospitals/Medical
5. Manufacturing
   a. Durable Goods
   b. Non-Durable Goods
6. Retail
7. Utilities
8. Wholesale

APPENDIX D

Job Titles in Each

Interviewed Business

American National Bank & Trust Co., Muncie. Finance
- Keypunch Operator
- Programming Manager
- Jr. Programmer
- Programmer

Ball Memorial Hospital, Muncie. Institutions, Hospital
- D.P. Center Manager
- Keypunch
- Operations Supervisor
- Operator
- Programmer

Ball State University, Muncie. Institutions, Colleges
- Data Entry Supervisor
- Sr. Data Entry Operator
- Data Entry Operator
- Sr. Operator
- Jr. Operator
- Operator Trainee
- Sr. Programmer
- Jr. Programmer
- Programmer Trainee
- Sr. Programmer Analyst
- Programmer Analyst
- Scheduler
- JCL Specialist

Warner Gear, Muncie. Manufacturing, Durable Goods

Data Entry

Keypunch

Operations Manager

Operator

Systems and Programming Coordinator

Programmer/Analyst

Jr. Programmer/Analyst

Sr. Programmer/Analyst

APPENDIX E

Tentatively Identified Tasks

keeps DP Center running smoothly
schedules employees
schedules meetings
attends/participates in meetings
handles unusual requests for programming
receives user requests for programming jobs
gives user requests to programmers
coordinates programmers
determines program priorities
appoints working teams
overall trouble shooter
participates in design of programs
receives programming assignments
writes programs for users' needs
compiles, debugs, and gets programs error-free
enters own program
does own program testing
does own trial runs on new programs
documents progrms
prepares flowcharts
up-dates programs
maintains programs
converts programs
modifies programs someone lese wrote
modifies purchased programs
writes program narrative
writes directions for program controls
writes directions for report distribution
implements changes to the system
problem analysis
keeps current programs on file

keeps a list of program changes
keeps a list of variable names
keeps sample reports by category
uses sample reports by category
keeps catalogue of files
uses catalogue of files
keeps catalogue of programs
uses catalogue of programs
responsible for own JCL
uses language's SPM
responsible if something goes wrong
reviews user output requirements
schedules/organizes computer usage
handles problems of whole system
librarian for tapes
coordinates operations and data entry
responsible for operators and data entry
knows daily routine
handles special requests from manager
utilizes written programs
picks up entered materials from keypunch/data entry
enters/inputs all data
changes ribbon on own machines
changes paper in own machines
changes disc mounts/packs
locates programs in files
locates correct disc packs
retrieves information on request
distributes materials to users
answers users' questions about machines
operates tape machine
operates card machine
operates burster
operates decollator
operates machine that keeps track of all CRTs
uses house standards for formating data

knows how to type-in formats
picks up unentered materials from user
sorts unentered materials
verifies someone else's work
reads cards with printing
sights cards without printing

Identified Skills

experience/knowledge of accounting is required
experience/knowledge of accounting is helpful
experience/knowledge in computer(s) language(s) required
experience/knowledge in computer(s) language(s) helpful
experience/knowledge in business required
experience/knowledge in business helpful
needs verbal communication skills
needss written communication skills
needs verbal and written communication skills
presentation skills required
presentation skills helpful
needs fast, accurate typing skills
needs moderate typing skills
needs sufficient efficient rate of strokes/hour
ability to be flexible is valuable

Tentative Activity Analysis
Computer Programmer, Small Computers

Uses programs previously written.
Updates, modifies, and revises previously written programs.
Writes new programs.

Programs may be for:

1. Business Applications
2. Engineering, Production
3. Statistics, Quality Control
4. Research & Development
5. Computer Assisted Instruction
6. Recreation (computer art, computer games)
7. Other

Programs are written in:

1. Basic
2. Fortran
3. APL
4. COBOL
5. RPG
6. PL/1
7. Assembler (BAL)
8. Pascal
9. Others

(A few companies have special languages--e.g., NCR's Neet 3.)

Levels of competence:

1. User cannot write programs, but has learned the regimen for using programs already written.
2. User knows a programming language; hence, can update, modify, and revise existing programs, and can originate

new programs to fit the unique needs of his particular organization.

3. User knows the assembler language and operating system of his computer.

Most users should be at level 2.

Vendors are encouraging the belief that "any clerk can use our computer," which is level 1, but users are finding that they need level 2 competence. Level 3 competence is beyond the daily needs and budget of most small businesses, although a level 3 consultant should be available.

APPENDIX G

Small Business--Small Computers

In a small business, one individual may be the whole computer department. There are several levels at which such an individual may interact with the computer.

At level 1, the person knows only the regimen (a series of steps) by which to:

a. Call for a specific program.
b. Input data into the program.
c. Get output data from the program.

For example, suppose the computer has a program to find the square root of any number. This program is named SQRRT. The person is inputting through a keyboard and the computer is outputting to a CRT (a Cathode Ray Tube, a TV screen which displays letters and numbers).

In the following interaction, what the computer displays is underlined; what the person types is not. (Systems differ slightly, but this example is typical of interactions in the BASIC language.)

HELLO
KEYWORD
KL21
OLD OR NEW?
OLD
OLD FILE NAME?
SSQRRT
READY
RUN

```
NUMBER
5
ANSWER = 2.23607
READY
BYE
```

Real programs, such as a General Ledger program, are much longer, with more steps and, often, various decision points. Nevertheless, this simple program illustrates two major points about level 1 competence:

a. The user can get the computer to run vendor supplied programs.
b. The user can do nothing else.

At level 2, the user knows the language of the program and can make any desired changes. For example, here is a program for getting a square root:

```
10 INPUT "NUMBER"; N
20 T = 1
30 A = N/T
40 T = (A + T)/2
50 IF (A - T<.000001) GOTO 70
60 GOTO 30
70 PRINT "ANSWER ="; A
80 END
```

As written, this program is good for the user who wants one answer, once in a while. But if he wants a second answer, he must go back, in the regimen, to entering OLD. So if the user will usually want several answers, he might add the following statements:

```
12 IF N<0 GOTO 80
75 GOTO 10
```

These will change the regimen in two ways:

a. After giving ANSWER = 2.23607, the computer will immediately request another number, NUMBER? The user need not (indeed, does not get the chance to!) type in OLD, SQRRT, and RUN.
b. The computer will no longer automatically terminate the run. To do so, the user must enter a negative number.

Note that the decision to revise the program, or not to, depends on the way the program is to be used, something the vendor cannot know.

If the user's numbers are generally less than 10, the program is good as given. But if they are more likely to be larger numbers, the user could add to the program:

```
15 FOR T = 1 TO N
20 IF T*T>N GOTO 30
25 NEXT N
```

(The new line 20 will replace the old line 20.)

This revision does not change the regimen, but it gets faster answers for large numbers. And again, this decision is something the vendor cannot know.

Vendors are usually supplying good general purpose programs, but just as usually, a general purpose program can be improved to fit a particular user's particular needs.

If such a small illustrative example program can occur in any one of four versions, it should be reasonable to realize that real (much longer) commercial programs can be more efficiently fitted to the user's needs in many various ways.

And this requires someone conversant with the language of the program.

It is for this reason that we recommend that every computer user have in house at least one person who has level 2 competence.

APPENDIX H

THE ONE PERSON D.P. DEPARTMENT
DEVELOPING A PROGRAM; A NARRATIVE DESCRIPTION

The boss calls with an idea about something he'd like done by computer. The DP person's first step is to clarify, rather precisely, just what kind of output the boss wants. From this, the DPer must decide what input data is required. If not all the required input data is available, then further discussion is needed until an achievable output is agreed upon.

This description assumes that, given a desired set of output data, the DPer quickly perceived both the required input data set and the required programming. This should be true for typical jobs, but for an unusual request, the DPer may need time to work out the requirements. To put it another way, an indispensable part of the DPer's job is either to already know, or else to work out, the set of instructions to the computer by which it can convert the available input data into the desired output data

The DPer must also write out the program adhering scrupulously to the syntax of the language, which often is BASIC. In a few instances, he may have a program already written which will handle the new application, or he may have a program which can be modified to do it, or he may have to originate an entirely new program.

When the program is written, it is then compiled and compiler errors are noted and corrected. It is then given a small sample of input data so devised as to encompass all the alternatives possibly encountered in real input data. (The boss may know of encounterable alternatives not conceived by

the DPer.) The resultant output data is calculated independently of the computer and compared with the computer output. Any discrepancies indicate logical errors which must be corrected. When these are corrected, the program is given real data which has been done in some way previously and the output checked. When the program is judged to be free of errors, it is documented. The documenting should include a printout listing of the program, a list of variables with brief descriptions of each, a flowchart, and a concise description of the program. If the program includes any unusual algorithms, these need to be described more fully. When the documenting is complete, the job of developing the program is complete. Such developed programs may often be reused, but such reuse becomes just a reliable routine.

APPENDIX I

THE ONE PERSON D.P. DEPARTMENT
DEVELOPING A PROGRAM; ACTIVITIES DESCRIPTION

1. Given a description of a desired output:

   a. If any part of the description is inadequate, prepare questions to elicit whatever further information is needed.

   b. When the output description is adequate, prepare a print chart of the desired output.

2. Given an adequate description of the desired output, tell completely what input is required--i.e., tell what files are needed, what records are in each file, what fields are in each record, how many characters are in each field, and what type of datum each field is (integers, reals, alphabetic, etc.)

3. Given an adequate description of the required input data:

   a. Know the name of and regimen for, or be able to write, a program to store the data on disk (or tape).

   b. Know the name of and regimen for, or be able to write, a program to copy the data from disk (or tape) into the user's program.

4. Given an adequate description of the I/O data, know the name of and regimen for, or be able to write, a program to produce the output data from the input data. (The "activity" of writing a program will itself be separated into a number of more detailed and specific activities.)

5. Given a written program, enter and run it.

6. Given a compiler error message, be able to make the required correction.

   a. Some messages require program corrections.

   b. Some messages require data entry corrections.

   c. Some messages are not immediately diagnosable and may require intelligent trial-and-error solution and/or use of TRACE techniques.

7. Given I/O data specifications, devise trial run input data which includes all the variations of inputs possible within the specifications.

8. Given a trial run input data, produce (independently of the computer) the resultant output data, making use of available cross-checking techniques.

9. Given trial run input data, run the program with it, getting trial run output data.

10. Given discrepancies between the independently calculated trial run output data and the program computed trial run output data be able to make the necessary corrections in the program.

11. Given a corrected program, make a list of variables, telling briefly the function (or functions) of each.

12. Given a corrected program, make a flowchart of it.

13. Given a corrected program, output a hard copy listing of it.

14. Given a corrected program, write a concise description of it.

15. Given a corrected program, be able to store it on disk (or tape):

    a. in the high level language.

    b. in machine language.

APPENDIX I2

Possible "Frame" For Organizing Tasks Into Domains

1. Given a description of a desired output:

   a. If any part of the description is inadequate, prepare questions to elicit whatever further information is needed.
   b. When the output description is adequate, prepare a print chart of the desired output.

2. Given an adequate description of the desired output, tell completely what input is required; i.e., tell what files are needed, what records are in each file, what fields are in each record, how many characters are in each field, and what type of datum each field is (integers, reals, alphabetic, sequential or random, etc.) and from what device each file will be input.

3. Given an adequate description of the required input data:

   a. Know the name of and regimen for, or be able to write, a program to store the data on disk (or tape).

   b. Know the name of and regimen for, or be able to write, a program to copy the data from disk (or tape) into the user's program.

4. Given an adequate description of the I/O data, know the name of and regimen for, or be able to write, a program to produce the output data from the input data. (The "activity" of writing a program will itself be separated into a number of more detailed and specific activities.)

5. Given a written program, enter and run it.

6. Given a compiler error message, be able to make the required correction.

   a. Some messages require program corrections.

   b. Some messages require data entry corrections.

   c. Some messages are not immediately diagnosable and may require intelligent trial-and-error solution and/or use of TRACE techniques.

7. Given I/O data specifications, devise trial run input data which includes all the variations of inputs possible within the specifications. (Check with boss for other possible variations.)

8. Given trial run input data, produce (independently of the computer) the resultant output data, making use of available cross-checking techniques.

9. Given trial run input data, run the program with it, getting trial run output data.

10. Given discrepancies between the independently calculated trial run output data and the program computed trial run output data, be able to make the necessary corrections in the program.

11. Given a corrected program, make a list of variables, telling briefly the function (or functions) of each.

12. Given a corrected program, make a flowchart of it.

13. Given a corrected program, output a hard copy listing of it.

14. Given a corrected program, write a concise description of it.

15. Given a correct program, be able to store it on disk (or tape):

    a. In the high-level language

    b. In machine language.

APPENDIX J

SELECTED TASKS FROM V-TECS LIST

TASK: Analyze company operations for beneficial use of the computer.

TASK: Handle special requests for data processing services.

TASK: Develop procedures relative to computer input and output.

TASK: Determine proper forms for printer.

TASK: Determine the forms needed for jobs.

TASK: Plan and schedule routine data services work assignments.

TASK: Consult with supervisors concerning problem areas.

TASK: Notify appropriate person or office of new or revised reporting requirements.

TASK: Determine destruction dates of computer files.

TASK: Determine priority of jobs to be run on the computer.

TASK: Assign priorities to computer jobs.

TASK: Select scheduling system for operations.

TASK: Monitor jobs processed and pending.

TASK: Monitor the meeting of deadlines.

TASK: Analyze documentation for completeness and accuracy.

TASK: Determine type of information needed to run a job.

TASK: Determine steps in running a job.

TASK: Control basic input into data processing system.

TASK: Code input data for computer entry.

TASK: Load Programs.

TASK: Analyze data processed to make sure that desired information is obtained.

TASK: Analyze data processed for modification and/or combination of reports.

TASK: Check output material to determine that work is complete.

TASK: Observe for verification printouts which indicate end of job or job steps.

TASK: Observe for abnormal termination of job.

TASK: Make periodic checks to clarify correctness of output.

TASK: Check error with appropriate staff and arrange for correction.

TASK: Determine cause of faulty output.

TASK: Arrange reruns to correct final output.

TASK: Distribute printed output to users.

TASK: Make recommendations on purchasing equipment.

TASK: Maintain records of machine performance.

TASK: Schedule preventive maintenance.

TASK: Power up computer and I/O devices.

TASK: Communicate with the computer through the console.

TASK: Set control switches on central processor.

TASK: Operate disk drive.

TASK: Initialize floppy disk.

TASK: Perform halt/restart procedures.

TASK: Load continuous forms in printer.

TASK: Remove output from machine.

TASK: Utilize operator's manuals.

TASK: Utilize utility program.

TASK: Run test of new program for system/programming.

TASK: Use a sort-merge utility program to sort records.

F	8	Write definition of input.
F	9	Write definition of output.
B	6	Determine input format for job.
B	9	Determine output format for job.
C	2	Determine logic sequence of programs.
C	1	Detail all specified calculations.
F	11	Select formulae for calculations.
B	11	Examine existing utility programs for application to a job.
B	10	Examine existing subroutines for application to a job.
D	16	Develop subroutines.
D	11	Code subroutines.
D	19	Code main line programs.
E	8	Desk check source programs for keypunching errors prior to assembly or compilation.
G	3	Correct program when "blow-up" occurs.
E	9	Desk check source program listing after assembly or compilation.

E	7	Design test data for use in program testing.
E	1	Analyze computer input for test run.
E	11	Test programs for accuracy.
E	2	Analyze computer output following test run.
A	12	Evaluate program test results.
E	10	Recommend solutions to undesirable conditions discovered during tests.
E	5	Correct programming errors discovered during testing.
E	6	Debug programs.
A	22	Review completed programs for accuracy.
E	16	Consult with user on test output.
A	28	Prepare orientation briefings for users.
A	13	Function as part of a team to solve user problems in information processing.
A	20	Recommend modification of systems to management.
A	21	Recommend modifidation of systems to users.
F	10	Write detailed description of data base.
C	5	Prepare detail flowchart.

E	13	Test programs for flexibility.
G	4	Modify new computer programs for increased flexibility.
A	16	Maintain stock levels of blank forms for programming.
A	15	Maintain programming library.
D	3	Code data-base update programs.
D	4	Code disk-merge programs.
D	5	Code memory-sort programs.
D	9	Code programs incorporating standard routines.
G	9	Update existing programs with changes.

APPENDIX L

DUTIES AND TASKS ANALYSIS

TASK A (PERFORMING PROGRAMMING SUPERVISORY FUNCTIONS)

1. Conduct on-the-job training sessions for programmers.
2. Consult with operations on adequate preparation of computer operating instructions.
3. Consult with systems designers on programming requirements of new system(s).
4. Develop programming aids.
5. Edit computer programs for efficient use of memory.
6. Establish standards for program documentation maintenance.
7. Establish standards for program maintenance.
8. Evaluate program documentation for clarity of instructions.
9. Evaluate program documentation for correct procedures.
10. Evaluate program documentation for proper form in relation to criteria.
11. Evaluate program test result.
12. Function as part of a team to solve user problems in information processing.
13. Identify content for on-the-job training in programming.
14. Maintain programming library.

15. Maintain stock of levels of blank forms for programming.

16. Orient newly assigned programmers.

17. Plan on-the-job training sessions in programming.

18. Prepare operational briefings.

19. Recommend modification of systems to management.

20. Recommend modification of systems to users.

21. Review completed programs for accuracy.

22. Review detail flowcharts for preparation of programs.

23. Review documentation of programs.

24. Review programs for adherence to programming standards.

25. Set documentation standards.

26. Set programming standards.

27. Prepare orientation briefings for users.

28. Consult with users on programming requirements of new systems.

TASK B (ANALYZING PROGRAMMING PROBLEMS)

1. Apply computer installation guidelines during analysis.

2. Apply organization guidelines during analysis.

3. Apply shop programming guidelines during analysis.

4. Consult technical references during analysis.

5. Design record layouts.

6. Determine input format for job.

7. Determine machine format for input.

8. Determine machine format for output.

9. Determine output format for job.

10. Examine existing subroutines for application to a job.

11. Examine existing utility programs for application to a job.

12. Select most efficient program for automating job.

TASK C (FLOWCHARTING SOLUTIONS TO PROGRAMMING PROBLEMS)

1. Detail all specified calculations.

2. Determine logic sequence of programs.

3. Flowchart data conversion.

4. Flowchart data input.

5. Prepare detailed flowcharts.

6. Prepare input-output operations flowchart for machine operators.

7. Prepare systems flowchart.

8. Verify flowchart(s) for solution to problem.

TASK D (CODING PROGRAMS)

1. Adapt programs to process on different computer configurations.

2. Code computer applications using a utility report program generator.

3. Code data-base update programs.

4. Code disk-merge programs.

5. Code memory-sort programs.

6. Code parameters to call disk-sort programs.

7. Code programs for inquiry device.

8. Code programs for simulation of test results as an audit.

9. Code programs incorporating standard routines.

10. Code software utility programs.

11. Code subroutines.

12. Code tape-merge programs.

13. Code tape-sort programs.

14. Convert numbers manually from one system to another.

15. Determine how data records are to be processed.

16. Develop subroutines.

17. Prepare program routines.

18. Select appropriate programming language for problem.

19. Code main line programs.

TASK E (TESTING PROGRAMS)

1. Analyze computer input for test run.

2. Analyze computer output following test run.

3. Analyze core dumps.

4. Analyze program processing efficiency during test run.

5. Correct programming errors discovered during testing.

6. Debug programs.

7. Design test data for use in program testing.

8. Desk check source programs for keypunching errors prior to assembly or compilation.

9. Desk check source program listing after assembly or compilation.

10 Recommend solutions to undesirable conditions discovered during tests.

11. Test program for accuracy.

12. Test program for efficiency.

13. Test program for flexibility.

14. Translate computer programs written in symbolic language into assembly language by non-computer means.

15. Utilize memory maps in testing.

16. Consult with user on test output.

TASK F (DOCUMENTING PROGRAMS)

1. Define work files.

2. Develop documentation narrative.

3. Develop listing of all programs.

4. Develop program job control statements.

5. Prepare console operator's run books with necessary instructions and messages.

6. Prepare control sheets for utility and library programs.

7. Write definition of input.

8. Write definition of output.

9. Write detailed description of data base.

10. Specify formulae for calculations.

11. Prepare final detailed flowcharts.

12. Prepare user's manual.

TASK G (PERFORMING PROGRAM MAINTENANCE)

1. Correct program error using documentation.

2. Correct program error without program documentation.

3. Correct program when "blow up" occurs.

4. Modify new computer programs for increased flexibility.

5. Provide instructions on program modification to operations personnel.

6. Regenerate operating system with changes.

7. Retest entire system when modification involves several programs.

8. Update documentation for programs.

9. Update existing programs with changes.

APPENDIX M

DUTIES AND TASK ANALYSIS

Classification

A Essential
B Important
C Of some use
D Questionable
F Not Needed

Some tasks are

Cle More clerical than D.P.
Man More managerial than D.P.
OJT Primarily in house training, on-the-job training
Adv Too advanced for two-year schools

Some tasks are practically identical, or predominently overlapping.

Some tasks, as worded, are specific to a shop, or a system. But the concept should be taught. (Possibly, the task should be reworded.)

TASK A (PERFORMING SUPERVISORY FUNCTIONS)

Rank	Comm. Class.			Ave.	Sig.
5	C	1.	Be able to talk to laymen.	5.67	1.70
14	C,Adv	2.	Consult with operations on adequate preparation of computer operating instructions.	12.33	2.49
12	C,Adv	3.	Develop programming aids.	12.00	0.82
15	OJT+C	4.	Edit computer programs for efficient use of memory.	14.33	0.47
1	A	5.	Evaluate program test results.	3.00	4.24
9	A	6.	Function as part of a team to solve user problems in information processing.	3.00	2.48
10	B	7.	Maintain programming library.	9.00	2.16
3	A	8.	Review completed programs for accuracy.	3.67	2.36
6	C	9.	Review detail flowcharts for preparation of programs.	6.00	2.45
8	A,Adv	10.	Review documentation of programs.	7.33	2.87

.	B,Adv	11.	Review programs for adherence to programming standards.	7.00	3.74
2	A+OJT	12.	Consult with users on programming requirements of new systems.	3.67	1.25
4	A	13.	Analyze company operations for beneficial use of the computer.	4.67	3.86
13	A+Cle	14.	Monitor the meeting of deadlines.	12.33	1.70
11	A	15.	Make recommendations on purchasing or leasing equipment.	10.00	3.74

TASK 8 (ANALYZING PROGRAMMING PROBLEMS)

14	A+OJT	1.	Apply computer installation guidelines during analysis.	10.67	6.18
11	A+OJT	2.	Apply organization guidelines during analysis.	9.67	6.18
10	A+OJT	3.	Apply shop programming guidelines during analysis.	9.67	4.78
8	B	4.	Consult technical references during analysis.	9.33	3.86
2	A	5.	Design record layouts.	5.33	1.70
1	A	6.	Determine input format for job.	4.33	2.49
3	A	7.	Determine output format for job.	5.33	2.49
9	A	8.	Examine existing subroutines for applications to a job.	9.33	5.31
6	A	9.	Examine existing utility programs.	7.67	2.62
13	A	10.	Select most efficient program for automating job.	10.33	4.11
4	A+Man	11.	Determine type of information needed to run a job.	5.67	4.64
5	A	12.	Determine steps in running a job.	6.67	4.64
12	A	13.	Control basic input into data processing system.	10.33	3.30
7	A	14.	Consult with supervisors concerning problem areas.	9.00	2.16
15	A	15.	Utilize utility program.	11.00	2.97

16	A	16.	Use a sort-merge utility program to sort records.	11.67	2.09

TASK C (FLOWCHARTING SOLUTIONS TO PROGRAMMING PROBLEMS)

3	A	1.	Detail all specified calculations.	4.00	.00
2	A	2.	Determine logic sequence of programs.	1.67	.94
4	A	3.	Flowchart data conversion.	4.00	2.16
1	A	4.	Flowchart data input.	1.67	.47
5	A	5.	Prepare detailed flowchart.	4.33	.47
7	A,OJT	6.	Prepare input-output operations flowchart for machine operators.	6.33	.94
6	A	7.	Verify flowchart(s) for solution to problems.	6.00	.00

TASK D (CODING PROGRAMS)

14	B	1.	Code data-base update programs.	12.00	2.83
10	B	2.	Code disk-merge programs.	10.67	1.25
8	B	3.	Code memory-sort programs.	9.67	2.49
7	B	4.	Code parameters to call disk-sort programs.	8.67	3.30
9	B	5.	Code programs for inquiry device.	10.33	2.87
6	B	6.	Code programs incorporating standard routines.	6.67	1.25
16	B	7.	Code program simulation of test results as an audit.	15.00	.00
11	B	8.	Code software utility programs.	10.67	4.50
5	B	9.	Code subroutines.	6.33	2.05
13	B	10.	Code tape-merge programs.	11.67	.47
12	B	11.	Code tape-sort programs.	11.00	1.41
15	D	12.	Convert numbers manually from one system to another.	12.00	5.66
2	A	13.	Determine how data records are to be processed.	2.67	2.36

4	A	14.	Develop subroutines	3.67 1.70
1	A	15.	Prepare program routines.	2.00 0.82
3	A	16.	Code main line programs.	3.00 .82

TASK E (TESTING PROGRAMS)

3	A	1.	Analyze computer input for test run.	3.67 1.25
5	A	2.	Analyze computer output following test run.	5.33 .47
20	F,Adv	3.	Analyze core dumps.	18.33 1.70
8	A	4.	Correct programming errors discovered during testing.	9.67- 2.87
6	A	5.	Debug programs.	7.00- 3.27
4	A	6.	Design test data for use in program testing.	3.67- 2.49
1	A	7.	Desk check source program for key-punching errors prior to assembly or compilation.	1.67- .94
2	A	8.	Desk check source program listing after assembly or compilation.	3.33 1.89
17	A	9.	Recommend solutions to undesirable conditions discovered during tests.	15.67 3.68
7	A	10.	Test program for accuracy.	7.67- .47
14	A	11.	Test program for flexibility.	13.67 3.40
19	Adv	12.	Utilize memory maps in testing.	18.00 .82
11	A	13.	Consult with user on test output.	11.33 .94
10	A	14.	Analyze data processed to make sure that desired information is obtained.	11.00 5.35
16	A+OJT	15.	Understand in-house checkpoints, benchmarks.	15.33 2.05
12	A	16.	Check output material to determine that work is complete.	11.33 2.87
15	A	17.	Make periodic checks to clarify correctness of output.	14.67 4.19
9	A	18.	Check program error with appropriate	10.67 4.71

			staff and arrange for program correction.		
13	A	19.	Determine cause of faculty output.	12.33	1.70
18	A	20.	Run test of new program for system/ programming.	15.67	4.78
TASK F (DOCUMENTING PROGRAMS)					
9		1.	Define work files.	9.33	3.40
6	A	2.	Develop documentation narrative.	5.00	2.83
8	A	3.	Develop listing of all programs.	9.33	.47
10	A	4.	Develop program job control statements.	11.00	2.94
16	C	5.	Prepare console operator's run books with necessary instructions and messages.	14.33	2.36
12	C	6.	Prepare control sheets for utility and library programs.	11.67	2.36
1	A	7.	Write definition of input.	1.27	.47
2		8.	Write definition of output.	3.00	.82
3	A	9.	Consult user on test output.	3.33	.47
7	A	10.	Write detailed description of data base.	8.33	4.78
5	A	11.	Specify formulae for calculations.	5.00	.00
4	A	12.	Prepare final detailed flowcharts.	4.67	2.62
17	A	13.	Prepare user's operating manual.	15.33	2.36
11	OJT+C	14.	Evaluate program documentation for clarity of instructions.	11.67	1.70
13	OJT+C	15.	Evaluate program documentation for correct procdures.	12.67	1.70
15	OJT+C	16.	Evaluate program documentation for proper form in relation to criteria.	13.67	1.70
14	A	17.	Analyze documentation for completeness and accuracy.	13.00	3.74
TASK G (PERFORMING PROGRAM MAINTENANCE)					
2	A	1.	Correct program error using documentation.	2.00	0.00

1	A	2.	Correct program error without program documentation.	1.67	.94
3	A	3.	Correct program when "blow up" occurs.	2.33	.94
5	B,Adv.	4.	Modify new computer for increased flexibility.	5.00	.82
6	A	5.	Update documentation for programs.	5.67	.47
4	A	6.	Update existing programs with changes.	4.33	.47

TASK H (OPERATING THE COMPUTER)

2	B+OJT	1.	Determine proper forms for printer.	4.00	1.63
1	B+OJT	2.	Determine the forms needed for jobs.	3.00	1.63
19	A+Man	3.	Plan and schedule routine data services work assignments.	17.33	10.14
25	A	4.	Consult with supervisors concerning problem areas.	22.67	2.05
27	A	5.	Notify appropriate person or office on new revised reporting requirements.	25.33	1.70
7	A+Man	6.	Determine priority of jobs to be run on the computer.	8.33	9.67
6	A+Man	7.	Determine priorities to computer jobs.	8.00	9.20
22	A+Man	8.	Select scheduling system for operations.	18.67	10.37
21	A	9.	Monitor jobs processed and pending.	18.00	6.38
26	A+Cle	10.	Monitor the meeting of deadlines.	25.33	1.25
4	A	11.	Code input data from computer entry.	7.00	3.27
10	A+OJT	12.	Load programs.	10.00	4.24
12	A	13.	Observe for verification printouts which indicate end of job or job steps.	12.67	5.44
17	A	14.	Observe for abnormal termination of job.	16.67	1.89
24	A	15.	Arrange reruns to correct final output.	21.00	2.45
20	A	16.	Record or save abnormal results.	17.67	1.89
23	B	17.	Maintain records of machine	20.33	3.09

			performance.		
15	A	18.	Be sensitive to abnormalities in computer operations.	16.00	1.41
3	A+OJT	19.	Power up computer and I/O devices.	6.33	1.25
9	A+OJT	20.	Communicate with the computer through the console.	9.33	4.03
5	A+OJT	21.	Set control switches on central processor.	8.00	.82
8	A+OJT	22.	Operate disk drive.	9.33	.47
12	A+OJT	23.	Initialize floppy disk.	13.00	5.10
14	A+OJT	24.	Perform halt/restart procedures.	15.67	5.10
11	A+OJT	25.	Load continuous forms in printer.	10.67	.94
16	A+OJT	26.	Remove output from machine.	16.33	4.50
18	A+OJT	27.	Utilize utility program.	17.33	3.86

APPENDIX M2

Tasks and Subtasks

A. Ascertain what output is desired and how it is to be set up.

A	1.	Determine output format for job.
B+OJT	2.	Determine the forms needed for jobs.
	3.	Write definition of output.
	4.	Fill out a printer spacing chart.
	5.	Explain the meaning of each of the symbols used on a printer spacing chart.

B. Tell what input is required to get the desired output, and how it is set up.

A+Man	1.	Determine type of information needed to run a job.
	2.	Define work files.
A	3.	Write detailed description of data base.
A	4.	Write definition of input.
A	5.	Determine input format for job.
A	6.	Flowchart data conversion.

A 7. Code input data for computer entry.

A 8. Flowchart data input.

A 9. Analyze computer input for test run.

A 10. Control basic input into data processing system.

C. Determine what steps (procedures, algorithms) are necessary to get the desired output from the required input.

A 1. Develop subroutines.

A+OJT 2. Apply computer installation guidelines during analysis.

B 3. Consult technical references during analysis.

A 4. Examine existing subroutines for applications to a job.

A 5. Select most efficient program for automating job.

A+OJT 6. Apply show programming guidelines during analysis.

A+OJT 7. Apply organization guidelines during analysis.

A 8. Design record layouts.

A 9. Determine steps in running a job.

A+OJT 10. Prepare input-output operations flowchart for machine operators.

A 11. Prepare program routines.

A 12. Examine existing utility programs for application to a job.

A 13. Determine logic sequence of programs.

A 14. Determine how data records are to be processed.

A 15. Specify formulae for calculations.

A 16. Detail all specified calculations.

D. Utilize, modify, combine, and/or originate programs in a high level language to accomplish the necessary steps.

B 1. Code data-base update programs.

A 2. Code main-line programs.

B 3. Code software utility programs.

B 4. Code programs incorporating standard routines.

B 5. Code programs for inquiry device.

B 6. Code program simulation of test results as an audit.

B 7. Code tape-merge programs.

A 8. Use a sort-merge utility program to sort records.

B 9. Code disk-merge programs.

B 10. Code memory-sort programs.

B 11. Code tape-sort programs.

A 12. Utilize utility program.

B 13. Code subroutines.

B 14. Code parameters to call disk-sort programs.

E. Find and correct, at any stage, all types of errors.

A 1. Verify flowchart(s) for solution to problems.

A 2. Debug programs.

A 3. Determine cause of faulty output.

A 4. Make periodic checks to clarify correctness of output.

A 5. Evaluate program test results.

A 6. Observe for abnormal termination of job.

A 7. Arrange reruns to correct final output.

A 8. Observe for verification printouts which indicate end of job or job steps.

A 9. Check program error with appropriate staff and arrange for program correction.

A 10. Correct program error using documentation.

A 11. Correct program error without program documentation.

A 12. Analyze data processed to make sure that desired information is obtained.

A 13. Test program for accuracy.

A 14. Check output material to determine that work is complete.

A 15. Review completed programs for accuracy.

A 16. Analyze documentation for completeness and accuracy.

A 17. Run test of new program for system/programming.

A+OJT 18. Understand inhouse checkpoints, benchmarks.

A 19. Design test data for use in program testing.

A 20. Consult with supervisors concerning problem areas.

A 21. Correct program when "blow up" occurs.

A 22. Analyze computer output following test run.

OJT+C 23. Evaluate program documentation for correct procedures.

A 24. Desk check source programs for keypunching errors prior to assembly or compilation.

A 25. Recommend solutions to undesirable conditions discovered during tests.

A 26. Correct programming errors discovered during testing.

A 27. Consult with supervisors concerning problem areas.

A 28. Desk check source program listing after assembly or compilation.

F,Adv 29. Analyze core dumps.

F. Prepare, modify, update, and/or maintain software

C 1. Prepare control sheets for utility and library programs.

A 2. Develop listing of all programs.

A 3. Develop documentation narrative.

OJT+C 4. Edit computer programs for efficient use of memory.

B,Adv 5. Modify new computer program for increased flexibility.

A 6. Update documentation for programs.

A 7. Prepare detail flowchart.

A 8. Update existing programs with changes.

A 9. Test program for flexibility.

OJT+C 10. Evaluate program documentation for clarity of instructions.

A 11. Record or save abnormal results.

B 12. Maintain programming library.

A 13. Prepare final detailed flowcharts.

A,Adv 14. Review documentation of programs.

A 15. Prepare user's operating manual.

OJT+C 16. Evaluate program documentation for proper form in relation to criteria.

C 17. Prepare console operator's run books with necessary instructions and messages.

B,Adv 18. Review programs for adherence to programming standards.

G. Update and maintain or obtain maintenance for hardware.

A 1. Be sensitive to abnormalities in computer operations.

B 2. Maintain records of machine performance.

A 3. Make recommendations on purchasing or leasing equipment.

H. Consult with other people.

C 1. Be able to talk to laymen.

C,Adv 2. Consult with operations on adequate preparation of computer operating instructions.

A+OJT 3. Consult with users on programming requirements of new systems.

A 4. Notify appropriate person or office of new revised reporting requirement.

A 5. Function as part of a team to solve user problems in information processing.

A 6. Consult with user on test output.

I. Schedule work.

A+Man 1. Determine priority of jobs to be run on the computer.

A+Man 2. Select scheduling system for operations.

A+Cle 3. Monitor the meeting of deadline.

A 4. Monitor jobs processed and pending.

A+Man 5. Plan and schedule routine data services work assignments.

J. Operate the equipment.

B+OJT 1. Determine proper forms for printer.

A+OJT 2. Initialize floppy disk.

A+OJT 3. Power up computer and I/O devices.

A+OJT 4. Operate disk drive.

A+OJT 5. Set control switches on central processor.

A+OJT 6. Load programs.

A+OJT 7. Communicate with the computer through the console.

A+OJT 8. Perform halt-restart procedures.

A+OJT 9. Load continuous forms in printer.

A+OJT 10. Remove output from machine.

K. Miscellaneous

A 1. Develop program job control statements.

C,Adv 2. Develop programming aids.

C 3. Review detail flowcharts for preparation of programs.

Adv 4. Utilize memory maps in testing.

D 5. Convert numbers manually from one system to another.

A,Man 6. Analyze company operations for beneficial use of the computer.

APPENDIX N

Data Processing With a Small Computer in a Small Company

Major Areas of Activity

1. Be able to tell what specific things the computer can do (cost efficiently) for the company.

2. Do all the steps needed to prepare data for input and to input it.

3. Do all the steps needed to process the data correctly.

4. Do all the steps needed to get the desired output.

5. Maintain, or obtain, maintenance for software and hardware.

Major Area #1

Decisions about the specific things the computer can do for the company will have already been made, either cooperatively or at a managerial level.

Major Area #2

1. List the data which must be input in order to produce the required output.

2. Show (list, chart) how the input data are to be organized into fields, records, and files.

3. Code the data for inputting.

4. Input the data.

Major Area #3

1. Be able to list all programs and subroutines available to do or to assist in doing a particular D.P. assignment.

2. List (or flowchart) the steps needed to go from the input data to the desired output.

3. Produce, in a high level language, the program needed.

   a. Use an available program.
   b. Modify an available program.
   c. Write a program using available subroutines.
   d. Write an original program.

4* Devise test data to simulate every possible event within the input data, and hand prepare test data output.

5* Input program and test data and run the program.

6. Correct compiler errors.

7* Compare the program output of test data with hand prepared output. Correct logic errors.

*These could also be listed under Major Area #4.

Major Area #4

1. Produce a printer spacing chart of the desired output.

2. Check the output format for conformity with the printer spacing chart. Make any needed corrections.

3. If anything occurs within the input data which has not already been tested for, the output should be hand calculated and checked and corrected, if necessary.

Note: Items 4, 5, and 7 of Major Area #3 could equally well be listed under Major Area #4.

To Data Processing With a Small Computer in a Small Company

1. Determine what output is desired, what input is required, and how to set up all of it.

2. Figure out what to do with the input data (what algorithms) to get the output data.

3. Write (or revise, or use) the program in a high level language. Execute it.

4. Find (at any stage) and correct all types of errors.

5. Prepare, modify, update, and maintain software. Update and maintain (or obtain maintenance for) hardware.

6. Operate all the equipment.

7. Be able to work with people.

APPENDIX O

Businesses Interviewed in 1979

Chrysler Corporation, New Castle. Manufacturing, Durable Goods

Citizens Federal Savings & Loan Association, New Castle. Finance

Diamond International Containers New Castle. Manufacturing, Non-Durable Goods

Gayee Sales, Inc., Muncie. Retail

Henry County Farm Bureau Co-Op, New Castle. Retail

Henry County Memorial Hospital, New Castle. Institutions, Hospitals

Ingersoll Johnson Steel Co., New Castle. Manufacturing, Durable Goods

Muncie Parts Manufacturing co., Inc., Muncie. Manufacturing, Durable Goods.

New Castle Community School Corp., New Castle. Institution, Schools

Reed Electric, New Castle. Utilities

3-M, Hartford City. Manufacturing, Non-Durable Goods

Systematics, Muncie. Retail

United Parts Co., New Castle. Retail

Businesses Interviewed in 1980

American National Bank & Trust Co., Muncie. Finance

Ball Corporation, Muncie. Manufacturing, Durable Goods

Ball Memorial Hospital, Muncie. Institutions, Hospitals/Medical

Ball State University. Muncie. Institutions, Colleges/Schools

Computerized Management Decisions, Inc., Anderson. Retail

Enterprise Computer Systems, Muncie. Retail

J.C. Penney, Inc., Muncie. Retail

Lift-A-Loft Corp., Muncie. Construction

Muncie Data Processing Center, Muncie. Institutions, Governmental

Warner Gear, Muncie. Manufacturing, Durable Goods

Warner Press, Inc., Anderson. Manufacturing, Non-Durable Goods

Westinghouse Electric Supply Co., Muncie. Wholesale

APPENDIX P

Brief Summary of Job Titles

Accounting Clerk

"Analyst" is apparently a rather general term.

Assistant (whatever) often do a lot of whatever they are assistant to.

Computer operator Console operator

Operator Computer operator leader Computer operator senior

Control

Control Clerk Control Operator

Data Entry

DE Clerk DE Specialist DE Supervisor Lead DE

Data Preparation Clerk Data Recorder DE Operator

Verifyer

"Director" of various things

"DP Manager" also seems rather general

EDP Manager EDP Supervisor

I/O (means input/output)

I/O Clerk I/O Control I/O Control Analyst

Encoder/Operator

Key, Keypunch (also see Data Entry)

Keypunch operator Keyboard operator

Operator/Data entry (also Data entry/operator)

Minicomputer

Operations

Op. Manager Op. Supervisor Op. Asst.

Payroll

Payroll clerk Payroll operator Payroll personnel

payroll/accounts clerk

Process engineer

Programmer

Programmer/analyst programmer/operator

senior programmer

Systems analyst

Systems programmer

Systems software manager

Tape librarian

Terminal operator

Unit record operator

APPENDIX Q

Annotated Listing of Job Titles

No. of Bus.	People (tot.)		
11	13	DP Supervisor	}Software?
6	6	DP Director	}Software?
3	4	DP Coordinator	}Hardware? or both?
34	42	DP Manager	}Hardware?
2	2	Operations Manager	
8	8	Operations Supervisor	
1	1	Operations Administrator	
1	1	Production Manager	
1	1	Night Terminal Manager	
2	7	Shift Supervisor	
4	11	Project Leader	
1	2	Project Manager	
1	1	Department Manager	
1	1	DP Supervisor/Programmer	}Should these be omitted as job titles since each title occurs separately elsewhere?
2	2	DP Manager/Programmer	
2	2	DP Manager/Analyst	
1	3	Computer Operator Supervisor	
3	5	Computer Operator Leader	
158	307	Computer Operator	
9	13	Sr. Computer Operator	
1	1	Jr. Operator	
1	3	Intermediate Console Operator	}Computer

		Operator?
7	15	Machine Operator)What machine? Keypunch?
1	2	Sr. Machine Operator Computer?
2	2	System Operator
2	2	Data Entry Supervisor
2	2	Data Entry Manager
2	2	Data Entry Leader
1	3	Data Entry Supervisor/Control Clerk
16	64	Data Entry Operator
122	261	Data Entry)Same as data entry operator?
5	5	Sr. Data Entry
2	8	Data Recorder)data entry?
1	2	Data Entry Specialist
1	1	Sr. Data Entry Specialist
2	2	Lead Keypunch Operator)Same as data entry?
34	114	Keypunch Operator)Same as data entry?
2	4	Sr. Keypunch Operator)Same as data entry?
3	3	Data Preparation
3	8	Encoder)Same as programmer?
1	3	Sr. Encoder)Same as programmer?
1	225	Data Transcriber)Same as data entry?
1	1	Order Checker
2	4	Order Entry
1	4	Unit Record Operator)Probably a keypunch (data entry) operator
1	2	Batch Control)Job scheduler?
1	1	Verifier
1	6	Proof Machine)Verifier?
1	1	Manager I/O
1	3	I/O Control
1	1	I/O Control Specialist)I/O means input/output

1	2	I/O Control Analyst	May all be the same
1	1	I/O Control Clerk	thing; the profile
1	1	I/O Clerk	similarity analysis might clarify this.
1	1	Control Manager	)Data control?
2	2	Control Supervisor	System control? CPU control?
1	1	Controls Coordinator	Etc.?
14	17	Control	
17	33	Control Clerk	
2	14	Keypunch Control	
1	1	Control Technician	
1	5	Data Control Specialist	
1	2	Systems Control Specialist	
1	1	Sr. Systems Control Specialist	
1	8	Productions Control	
1	3	Interior Control)?	
2	2	Systems & Programming Manager	
2	2	Systems/Programming Supervisor	
4	4	Systems Manager)Software? Hardware?	
1	1	Systems Software Manager	
1	1	Systems Designer)Software? Hardware?	
1	1	Information Systems Designer	
1	1	Scientific Systems Designer	
1	1	Programmer/System Design	
19	74	Systems Analyst	
2	2	Sr. Systems Analyst	
1	1	Technical Analyst	
1	4	Scientific Systems Analyst	
1	1	Software Support Analyst	
29	48	Analyst	
3	5	Sr. Analyst	

1	1	Jr. Analyst
1	1	Scientific Information Analyst
48	122	Programmer/Analyst
3	7	Sr. Programmer/Analyst
1	1	Programmer/Analyst Intermediate
78	158	Programmer
8	11	Sr. Programmer
2	6	Jr. Programmer
4	4	Programmer Operator
2	4	Programmer/Operation Supervisor
1	1	Chief Clerk
1	1	Clerk
1	3	Administrative Clerk
1	1	Communication Clerk
2	2	Quality Control Clerk
2	2	Quality Control Clerk
6	37	DP Clerk
1	8	Data/Code Clerk)?
1	1	Renew Center Clerk)?
1	1	Document Clerk
1	1	Utilities Clerk
1	1	Billing Clerk
2	3	Accounting Clerk
2	3	Balance clerk
1	1	Customer Account Clerk
1	1	Payroll Accounts Payable Clerk
1	1	Users Account Pay
1	5	Users Account Rec
1	2	Users Payroll
1	2	Bookkeeper
1	1	Head of Billing Department
2	2	Manager Information Systems
1	18	Information Systems Member)Manager?
1	5	Information Systems Sr. Member)Manger?
4	4	Scheduler, Computer (Job)
1	2	Advanced Editor)Very vague

1	1	Data Services Specialist
1	2	Alert Systems Engr.]Very vague
1	1	JCL Specialist
1	1	Consultant
1	1	Services Staff Specialist]Very vague
3	11	Computer Technicians
4	6	Librarian

APPENDIX B

A Listing of Time Patterns of Two Job Titles

Description	No.	Percent of Time Spent: Sys. Dsgn.	Prog.	Mach. Op.	Control	Other	Sim. to Ave.	Sim. to # 1.
Analyst/Programmer	01	50	50	00	00	00	.90	1.00
	02	20	80	00	00	00	0.70	0.47
	03	90	10	00	00	00	-.01	0.22
	04	50	50	00	00	00	0.90	1.00
	05	40	40	00	00	20	0.83	0.79
	06	40	60	00	00	00	0.97	0.92
	07	10	90	00	00	00	0.46	0.32
	08	40	10	00	00	50	0.10	0.09
Programmer/Analyst	09	50	50	00	00	00	0.90	1.00
	10	25	65	00	00	10	0.91	0.64
	11	10	60	00	00	30	0.59	0.32
	12	40	60	00	00	00	0.97	0.92
	13	40	50	00	00	10	0.97	0.92
	14	30	70	00	00	00	0.90	0.72
	15	05	95	00	00	00	0.33	0.10
	16	30	70	00	00	00	0.90	0.72
	17	50	50	00	00	00	0.90	1.00
	18	30	70	00	00	00	0.90	0.72
	19	30	70	00	00	00	0.90	0.72
	20	30	70	00	00	00	0.90	0.72
	21	07	42	34	00	17	0.39	0.20
	22	35	30	10	10	15	0.89	0.65
	23	50	50	00	00	00	0.90	1.00
	24	50	50	00	00	00	0.90	1.00
	25	50	45	00	00	05	0.88	0.98
Ave.		36	56	02	00	06	1.00	0.90
Sig.		18	21	07	02	12		

APPENDIX S

Pattern Similarity Coefficient

Referenced under its originator: Cattell, Raymond B.

Given two sets of data:
No. 1: A(1), B(1), C(1), . . . N(1)
No. 2: A(2), B(2), C(2), . . . N(2)

Then let d(1) = A(1) - A(2), d(2) = B(1) - B(2)
d(3) = C(1) - C(2) . . . d(N) = N(1) - N(2).

And let $d^2 = (d(1))2 + (d(2))2 + \ldots, (d(N))2$

Then $r_p = \frac{2K - \Sigma d^2}{2K + \Sigma d^2}$

where 2K is the most probable random chance value for Σd^2. That is, a degree of similarity due only to random chance should produce an average r_p of 0.00.

When Σd^2 is the sum of the squared standard score differences, then K is the median of chi-square value for K degrees of freedom.

Appendix T

Matrix of Pattern Similarity Coefficients for Analyst/Programmer and Programmer/Analyst

Item	No 01	No 02	No 03	No 04	No 05	No 06	No 07	No 08	No 09
	No 10	No 11	No 12	No 13	No 14	No 15	No 16	No 17	No 18
	No 19	No 20	No 21	No 22	No 23	No 24	No 25		
No 01		0.47	0.22	1.00	0.79	0.92	0.22	0.89	1.00
	0.68	0.32	0.92	0.92	0.72	0.10	0.72	1.00	0.72
	0.72	0.72	0.20	0.65	1.00	1.00	0.93		
No 02	0.47		-0.32	0.47	0.35	0.72	0.92	-0.22	0.47
	0.87	0.56	0.72	0.56	0.92	0.83	0.92	0.47	0.92
	0.92	0.92	0.24	0.23	0.47	0.47	0.40		
No 03	0.22	0.32		0.22	0.14	0.00	0.44	0.00	0.22
	-0.19	-0.32	0.00	0.09	-0.18	-0.49	-0.18	0.22	-0.18
	-0.18	-0.18	-0.30	0.13	0.22	0.22	0.27		
No 04	1.00	0.47	0.22		0.79	0.92	0.22	0.89	1.00
	0.68	0.32	0.92	0.92	0.72	0.10	0.72	1.00	0.72
	0.72	0.72	0.20	0.65	1.00	1.00	0.93		
No 05	0.79	0.35	0.14	0.79		0.72	0.14	0.47	0.79
	0.68	0.56	0.72	0.92	0.56	0.84	0.56	0.79	0.56
	0.56	0.56	0.38	0.87	0.79	0.79	0.87		
No 06	0.92	0.72	0.00	0.92	0.72		0.47	0.00	0.92
	0.87	0.47	1.00	0.92	0.92	0.34	0.92	0.92	0.92
	0.92	0.92	0.27	0.57	0.92	0.92	0.87		
No 07	0.22	0.92	-0.44	0.22	0.14	0.47		0.32	0.22
	0.68	0.47	0.47	0.32	0.72	0.98	0.72	0.22	0.72
	0.72	0.72	0.14	0.04	0.22	0.22	0.16		

No 08	0.09	-0.22	0.00	0.09	0.47	0.00	-0.32		0.09
	0.02	0.14	0.00	0.22	-0.11	-0.37	-0.11	0.09	-0.11
	-0.11	-0.11	0.07	0.45	0.09	0.09	0.09	0.20	
No 09	1.00	0.47	0.22	1.00	0.79	0.92	0.22	0.09	0.68
	0.32	0.92	0.92	0.72	0.19	0.72	1.00	0.72	0.72
	1.00	0.72	0.72	0.65	1.00	1.00	0.98		
No 10	0.68	0.87	-0.19	0.68	0.68	0.87	0.68	0.02	0.68
		0.77	0.87	0.83	0.94	0.56	0.94	0.68	0.94
	0.94	0.94	0.42	0.53	0.68	0.68	0.65		
No 11	0.32	0.56	-0.32	0.32	0.56	0.47	0.47	0.14	0.32
	0.77		0.47	0.56	0.56	0.49	0.56	0.32	0.56
	0.56	0.56	0.50	0.44	0.32	0.32	0.34		
No 12	0.92	0.72	0.00	0.92	0.72	1.00	0.47	0.00	0.92
	0.87	0.47		0.92	0.92	0.34	0.92	0.92	0.92
	0.92	0.92	0.92	0.57	0.92	0.92	0.87		
No 13	0.92	0.56	0.09	0.92	-0.92	0.92	0.32	0.22	0.92
	0.83	0.56	0.92		0.79	0.20	0.79	0.92	0.79
	0.79	0.79	0.36	0.77	0.92	0.92	0.94		
No 14	0.72	0.92	-0.18	0.72	0.56	0.92	0.72	-0.11	0.72
	0.94	0.56	0.92	0.79		0.60	1.00	0.72	1.00
	1.00	1.00	0.29	0.42	0.72	0.72	0.65		
No 15	0.10	0.83	-0.49	0.10	0.04	0.34	0.98	-0.37	0.10
	0.56	0.40	0.34	0.20	0.60		0.60	0.10	0.60
	0.60	0.60	0.08	-0.05	0.10	0.10	0.50		
No 16	0.72	0.92	-0.18	0.72	0.56	0.92	0.72	-0.11	0.72
	0.94	0.56	0.92	0.79	1.00	0.60		0.72	1.00
	1.00	1.00	0.29	0.42	0.72	0.72	0.65		

o 17	1.00	0.47	0.22	1.00	0.79	0.92	0.22	0.09	1.00
	0.68	0.32	0.92	0.92	0.72	0.10	0.72	0.10	0.72
	0.72	0.72	0.20	0.65	1.00	1.00	0.98		
o 18	0.72	0.92	-0.18	0.72	0.56	0.92	0.72	-0.11	0.72
	0.94	0.56	0.92	0.79	1.00	0.60	1.00	0.72	
	1.00	1.00	0.29	0.42	0.72	0.72	0.65		
o 19	0.72	0.92	-0.18	0.72	0.56	0.92	0.72	-0.11	0.72
	0.94	0.56	0.92	0.79	1.00	0.60	1.00	0.72	1.00
		1.00	0.29	0.42	0.72	0.72	0.65		
o 20	0.72	0.92	-0.18	0.72	0.56	0.92	0.72	-0.11	0.72
	0.94	0.56	0.92	0.79	1.00	0.60	1.00	0.72	1.00
	1.00		0.29	0.42	0.72	0.72	0.65		
o 21	0.20	0.24	-0.38	0.20	0.38	0.27	0.14	0.07	0.20
	0.42	0.50	0.27	0.36	0.29	0.08	0.29	0.20	0.29
	0.29	0.29		0.51	0.20	0.20	0.23		
o 22	0.65	0.23	0.13	0.65	0.87	0.57	0.04	0.46	0.65
	0.53	0.44	0.57	0.77	0.42	-0.05	0.42	0.65	0.42
	0.42	0.42	0.51		0.65	0.65	0.74		
o 23	1.00	0.47	0.22	1.00	0.79	0.92	0.22	0.09	1.00
	0.68	0.32	0.92	0.92	0.72	0.72	0.10	0.72	1.00
	0.72	0.72	0.20	0.65		1.00	0.98		
o 24	1.00	0.47	0.22	1.00	0.79	0.92	0.22	0.09	1.00
	0.68	0.32	0.92	0.92	0.72	0.72	0.10	0.72	1.00
	0.72	0.72	0.20	0.65	1.00		0.98		
o 25	0.93	-.40	0.27	0.98	0.87	0.87	0.16	0.20	0.98
	0.65	0.34	0.87	0.94	0.65	0.05	0.65	0.98	0.65
	0.65	0.65	0.23	0.74	0.98	0.98			

APPENDIX U

Brief Cluster Analysis of

Pattern Similarity Coefficient Matrix

Cluster I:

Items 1, 4, 9, 17, 23, 24: All are 1.00 to each other.
Item 25 is .98 to all items of cluster I.
Items 6, 12, 13 are all .92 to all items of cluster I.
Items 13, 25 are .94 to each other.

Cluster II:

Items 14, 16, 18, 19, 20: All are 1.00 to each other.
Item 10 is .94 to all items of Cluster II.
Items 2, 6, 12 are all .92 to all items of cluster II.

All items of cluster I are .72 to all items of cluster II.

Other pairs:	r(p):
6, 12	1.00
7, 15	.98
2, 7	.92
5, 13	.92
6, 13	.92

XII. Final Report Abstract

A. PRODUCTS:

1. A teaching unit, Program Writing in Ten Tasks.
2. A pilot study of time-spent pattern similarity coefficients among job titles.

B. OBJECTIVES:

1. To develop a performance-based instructional program.
2. To identify the DP tasks within various job titles.

C. PROCEDURES:

1. Task lists from various sources were sorted and merged into a final detailed list of tasks and subtasks, under the guidance of an advisory committee, resulting in Program Writing in Ten Tasks.
2. A pilot study found Objective 2 to be difficult of achievement unless terms being used were further clarified, and suggested an empirical method for doing so.

D. OUTCOMES:

1. Better quality and more uniformity in the teaching of programming courses.
2. Better quality and more uniformity in the defining of job titles used in industry.